Digital Communications and Networks 9 (2023) 827–835

Contents lists available at ScienceDirect

Digital Communications and Networks

journal homepage: www.keaipublishing.com/dcan

Monocular 3D object detection with Pseudo-LiDAR confidence sampling and hierarchical geometric feature extraction in 6G network

Jianlong Zhang [a], Guangzu Fang [a], Bin Wang [a,**], Xiaobo Zhou [b], Qingqi Pei [c], Chen Chen [c,*]

[a] School of Electronic Engineering, Xidian University, Xi'an, 710071, China
[b] School of Computer Science and Technology, College of Intelligence and Computing, Tianjin University, Tianjin, 300000, China
[c] School of Telecommunications Engineering, Xidian University, Xi'an, 710071, China

ARTICLE INFO

Keywords:
Monocular 3D object detection
Pseudo-LiDAR
Confidence sampling
Hierarchical geometric feature extraction

ABSTRACT

The high bandwidth and low latency of 6G network technology enable the successful application of monocular 3D object detection on vehicle platforms. Monocular 3D-object-detection-based Pseudo-LiDAR is a low-cost, low-power solution compared to LiDAR solutions in the field of autonomous driving. However, this technique has some problems, i.e., (1) the poor quality of generated Pseudo-LiDAR point clouds resulting from the nonlinear error distribution of monocular depth estimation and (2) the weak representation capability of point cloud features due to the neglected global geometric structure features of point clouds existing in LiDAR-based 3D detection networks. Therefore, we proposed a Pseudo-LiDAR confidence sampling strategy and a hierarchical geometric feature extraction module for monocular 3D object detection. We first designed a point cloud confidence sampling strategy based on a 3D Gaussian distribution to assign small confidence to the points with great error in depth estimation and filter them out according to the confidence. Then, we present a hierarchical geometric feature extraction module by aggregating the local neighborhood features and a dual transformer to capture the global geometric features in the point cloud. Finally, our detection framework is based on Point-Voxel-RCNN (PV-RCNN) with high-quality Pseudo-LiDAR and enriched geometric features as input. From the experimental results, our method achieves satisfactory results in monocular 3D object detection.

1. Introduction

Recently, with the rapid development of technology, e.g., the Internet of Things [1,2], the Internet of Vehicles [3,4], big data and deep learning [5,6], urban transportation has also moved toward modernization and intelligence. Autonomous driving is one of the most promising tasks in intelligent transportation [7]. With the successful application of 6th-Generation (6G) cellular networks, autonomous driving has become one of the core services of 6G [8]. 6G can provide low-latency communication capability for autonomous driving, increase the synergy between vehicles and roads, and promote the rapid development of autonomous driving. The core challenge of autonomous driving is environmental perception. With the successful application of 2D object detection to the environmental perception of vehicles [9], 3D object detection has received increasing attention from scholars. 3D object detection can obtain the class, pose and precise position information of an object, which makes it widely used in many fields.

Existing 3D object detection methods mainly use LiDAR, cameras [10] and multisensor fusion schemes to collect 3D data. LiDAR-based 3D object detection has the best performance due to the accurate 3D point cloud provided by LiDAR. However, the quality of the generated LiDAR point cloud depends heavily on the laser scan and is limited by the material and process, which makes the cost very high. In the search for a lower-cost 3D object detection solutions, people are starting to use cameras instead of LiDAR to cut costs. Monocular 3D (M3D) object detection is promising compared to LiDAR-based detection methods, but it is more challenging without depth information. Although many associated studies [11,12] have made great progress, there are still problems to be solved in monocular 3D object detection.

Previous methods obtain 3D parameters in 3D space by applying 2D detection methods in image space. MonoGRNet [13] predicts the parameters of 3D bounding boxes in a single image according to the results

* Corresponding author.
** Corresponding author.
E-mail addresses: jlzhang@mail.xidian.edu.cn (J. Zhang), gzfang@stu.xidian.edu.cn (G. Fang), bwang@xidian.edu.cn (B. Wang), xiaobo.zhou@tju.edu.cn (X. Zhou), qqpei@mail.xidian.edu.cn (Q. Pei), cc2000@mail.xidian.edu.cn (C. Chen).

https://doi.org/10.1016/j.dcan.2022.05.003
Received 14 October 2021; Received in revised form 15 April 2022; Accepted 10 May 2022
Available online 16 May 2022

of 2D detection. M3D-Region Proposal Network (RPN) [14] utilizes PRN structures to optimize both 2D and 3D boxes. These methods cannot achieve better performance due to the lack of 3D spatial information, although they attempt to take some valid prior knowledge or reasonable constraints of image appearance. In contrast, the monocular Pseudo-LiDAR method makes full use of the LiDAR-based detection network by converting the visual point cloud to a Velodyne LiDAR point cloud and achieves excellent detection performance. Thus, recently, some approaches generated point clouds through monocular depth estimation networks and directly predicted 3D bounding boxes on 3D point clouds instead of learning 3D parameters in monocular images. Pseudo-LiDAR [15] uses the Frustum PointNet 3D detection framework [16] in 3D space to predict 3D bounding boxes of objects.

However, the Pseudo-LiDAR obtained by monocular depth map transformation is different from the LiDAR in terms of density and depth. The monocular depth estimation is usually affected by the background at the edge of the object vehicle with a large estimation error [17]; meanwhile, the error increases with the scene distance. From Fig. 1, we can observe that the PV-RCNN network fails to detect the long-distance objects in the raw Pseudo-LiDAR. This is also limited by the existing feature extraction structure. The number of points belonging to long-distance objects is very low, so the existing feature extraction structure has difficulty effectively representing the geometric features of the point cloud.

Herein, monocular Pseudo-LiDAR 3D object detection with a confidence sampling strategy and a hierarchical geometric feature extraction module is proposed to address the aforementioned problems. The main idea of our approach is to improve the quality of Pseudo-LiDAR and improve the geometric feature representation of point clouds. First, we propose a Pseudo-LiDAR confidence sampling strategy that fits the confidence distribution of Pseudo-LiDAR by a 3D Gaussian function to assign small confidence to points at the edges of object vehicles and at long distances and filter out points with larger depth estimation errors (i.e., points with small confidence) according to the confidence sampling. Then, we introduce a LiDAR-based 3D detection network PV-RCNN [18] as the 3D detector of our 3D detection framework. Finally, we design a hierarchical geometric feature extraction module that uses local attention feature encoding to construct the local neighborhood of the point cloud and aggregates the features by attention weights. It uses a dual transformer built with scalar and vector attention to capture the global correlation between points. Thus, the module can extract local geometric neighborhood features and global geometric features of Pseudo-LiDAR. Compared with the PointNet-based set abstraction method in PV-RCNN [18], our feature extraction module is more effective in extracting the geometric features of Pseudo-LiDAR.

The contributions of this paper can be summarized as follows.

- The benchmark of monocular 3D object detection is improved by employing the PV-RCNN network for the 3D detection framework.
- A confidence sampling strategy based on a 3D Gaussian distribution is designed to greatly reduce the impact caused by monocular depth estimation errors.
- Our model can obtain the geometric features of the point cloud extracted by the designed hierarchical geometric feature extraction module.

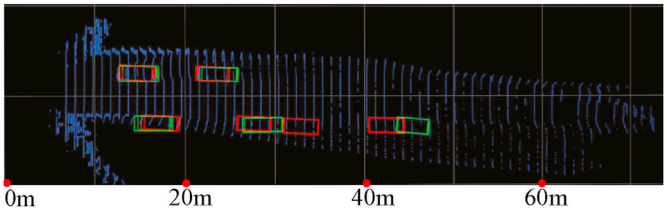


Fig. 1. PV-RCNN [18] network obtains KITTI [19] 3D detection results on raw Pseudo-LiDAR. Green boxes: ground truth. Red boxes: predicted.

The remainder of the paper is organized as follows. In Section 2, we present related work on 3D object detection. Section 3 shows the principles of our method. The relevant experiments are reviewed in Section 4. We summarize our work in Section 5.

2. Related work

2.1. LiDAR-based 3D object detection

Our inspiration comes from LiDAR-based 3D object detection. The LiDAR point cloud is simple and accurate compared with image data with RGB color information and possesses more advantages for object localization and pose estimation. Therefore, processing the point cloud data becomes the kernel task of the 3D object detection algorithm. These methods can be categorized into point-based and voxel-based methods. The first method uses the point cloud format to represent 3D scenes, and the main studies of this method are as follows. Frustum PointNet [16] projects the 2D detection box into a 3D frustum proposal and then uses PointNet [20] to segment the foreground and background of the frustum proposal to attenuate the interference of background points. PointRCNN [21] segments the foreground and background of the scene using PointNet++ [22] to generate a high-quality 3D proposal and then optimizes the proposal based on the pooled features of the point cloud.

The other method converts the point cloud into voxels to extract features from the voxels for 3D object detection. For example, SECOND [23] improves the point cloud feature extraction ability and accelerates the speed of the training by using a spatially sparse convolutional network that fully exploits the sparsity of the point cloud. The fast Point R–CNN [24] generates a small number of proposals through the Voxel-RPN layer and performs a more fine-grained regression on each proposal. PV-RCNN [18] generates high-quality proposals by the voxel-based approach, enriches the local features of the point cloud by the point-based approach, and further refines the generated proposals. While LiDAR-based 3D object detection achieves promising detection performance, collecting high-quality LiDAR point clouds is very expensive.

2.2. Monocular 3D object detection

Unlike LiDAR-based object detection methods, monocular 3D object detection utilizes a single image without 3D spatial information, which is a significant challenge in 3D detection tasks. In earlier works, mature 2D detectors were adopted to predict the 3D bounding box, and then the parameters of the bounding box were optimized with the geometric constraint. Mono3D [25] utilizes prior knowledge to extract 3D proposals in 3D space and projects them into 2D image space. Then, it uses the features within the projected box for SVM classification and further fine-tunes the proposal. Deep MANTA [26] obtains information of the 3D bounding box by detecting the key points in the vehicle feature areas. RoI-10D [27] introduces a differentiable 2D-RoI lift structure and fuses RGB images and monocular depth features by 2D-RoI to obtain 3D bounding boxes. Despite some improvement in 3D object detection performance achieved by using geometric relationship constraints or depth information, these methods are still limited by the lack of corresponding 3D spatial information.

Benefiting from the rapid development of the monocular depth estimation network, one can obtain abundant monocular depth information. Therefore, we transform the monocular depth map into a Pseudo-LiDAR point cloud to represent the 3D spatial information. AM3D [28] captures the object vehicle regions on the depth map by using a 2D detection network, projects the object vehicles from the 2D image into the point cloud with color information, and regresses the 3D bounding boxes with PointNet. Pseudo-LiDAR [15] directly transforms the obtained depth map from the depth estimation network into Pseudo-LiDAR. LiDAR--based 3D object detection methods can be applied to Pseudo-LiDAR, e.g., Frustum PointNet [16]. RefinedMPL [29] uses a supervised and an

unsupervised method to distinguish the foreground points and then filters out the interfering background points to address the situation in which the high density of Pseudo-LiDAR point clouds and many background points near image edges interfere with the detection. The associated experiments show excellent detection performance on monocular images.

Monocular Pseudo-LiDAR 3D object detection transforms the 2D feature extraction problem in monocular images into the problems of monocular depth estimation and point cloud feature extraction. Better monocular 3D detection performance is supposed to be obtained by using a mature monocular depth estimation network and a high-performance LiDAR-based 3D object detection network. It was verified in RefinedMPL [29] that the background points influence the detection performance. Therefore, to improve the generation quality of Pseudo-LiDAR and optimize existing LiDAR-based 3D object detection networks, we proposed a novel monocular 3D object detection method.

3. Approach

We proposed a monocular Pseudo-LiDAR 3D detection framework for obtaining 3D bounding boxes from monocular images. Existing 3D detection methods for monocular Pseudo-LiDAR, e.g., RefinedMPL [29], mainly improve the detection performance by filtering a large number of background points and utilizing a high-performance LiDAR-based 3D detection network. In most cases, the high-quality point clouds can reduce the computational cost of the 3D detector and generate high-quality proposals, which is why the existing LiDAR-based 3D detection networks can achieve excellent detection performance in monocular Pseudo-LiDAR 3D object detection.

Our proposed method applies a Pseudo-LiDAR confidence sampling strategy and a hierarchical geometric feature extraction module to improve the quality of the generated Pseudo-LiDAR point clouds and enhance the detection performance with the LiDAR-based detection network. As shown in Fig. 2, our approach has three stages. In the Pseudo-LiDAR generation stage, we employ DORN [30] to obtain depth information. We obtain Pseudo-LiDAR from the obtained depth estimation map by using coordinate transformation and projection transformation. In Pseudo-LiDAR confidence sampling, we use confidence sampling to generate high-quality Pseudo-LiDAR point clouds. First, we fit the local and global confidence distribution of the Pseudo-LiDAR according to the scene distribution. Then, the points with large monocular depth estimation errors are filtered out by confidence sampling to obtain higher quality Pseudo-LiDAR point clouds. In the third stage, i.e., Pseudo-LiDAR object detection, we design a hierarchical geometric feature extraction module to obtain satisfactory detection performance.

In this stage, we use PV-RCNN as our backbone network. Then, to address the problem of weak geometric feature representation capability for keypoints in PV-RCNN [18], we use the local attention feature encoding and the dual transformer to acquire the local and global geometric features of the Pseudo-LiDAR, respectively. Finally, the final detection results are obtained by the bounding box prediction network of the PV-RCNN.

3.1. Pseudo-LiDAR generation

3.1.1. Monocular depth estimation

The monocular depth estimation network obtains pixelwise depth information output by using the image as input. Because the accuracy of depth estimation determines the quality of the Pseudo-LiDAR point clouds, we adopt the successful DORN [30] as our monocular depth estimation network to obtain the depth map D. The DORN algorithmic framework introduces ordered regression to solve the problem of less accurate depth point estimation for point clouds [30].

3.1.2. Pseudo-LiDAR generation

The depth map D only obtains the depth information of the RGB image space. Depth information needs to be further obtained by coordinate transformation and projection transformation to generate the Pseudo-LiDAR point cloud. The process is shown in Fig. 2. The depth map is in a discrete pixel coordinate system, and the Pseudo-LiDAR is in a continuous physical coordinate system. The 3D coordinates (x, y, z) of the Pseudo-LiDAR point cloud are obtained by applying a projection transformation [15] to each pixel coordinate (u, v) according to the following equations:

$$\begin{cases} x = \dfrac{z}{f_c}x', x' = s_x(u - u_0) \\ y = \dfrac{z}{f_c}y', y' = s_y(v - v_0) \\ z = D(x', y') \end{cases} \tag{1}$$

where x' and y' are the coordinates of the physical distance, s_x and s_y are the physical distances between two arbitrary pixels, and (u_0, v_0) is the position of the origin of the physical coordinate system in the pixel coordinate system. f_c is the camera focal length.

3.2. Confidence sampling

In the Pseudo-LiDAR generation process, the depth information is encoded into the point cloud. However, the limitations of the monocular depth estimation network result in large depth estimation errors both at

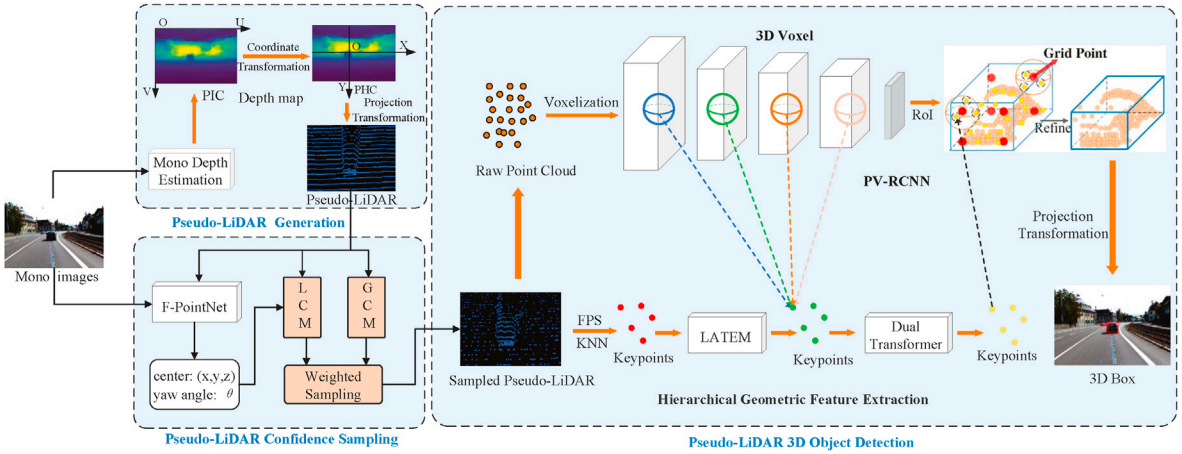


Fig. 2. Overall architecture of our framework. PIC: Pixel Coordinate System; PHC: Physical Coordinate System; FPS: Farthest Point Sampling; LATEM: Local Attention Feature Encoding Module.

the edges of the object vehicle and in regions with long distances. To reduce the influence of the depth estimation error, we first assign the corresponding confidence to each point. We calculate the local confidence based on the distance between the points and the 3D center point of the object vehicle. Then, the global confidence is obtained according to the distance distribution of the scene. Finally, Pseudo-LiDAR confidence is obtained by weighting the global confidence with the local confidence, and the point cloud is sampled according to the confidence distribution.

3.2.1. 3D center point and direction estimation

The local confidence requires the distribution of the distance between the points and the 3D center point inside the object vehicle. Therefore, we need to know the yaw angle and 3D center point of the object vehicle to determine whether the point is inside the object vehicle and to know the distance between points. We use the F-PointNet network to estimate the 3D center point (x_c, y_c, z_c) and yaw angle θ of the object vehicle. F-PointNet [16] generates a 3D Frustum proposal by a 2D CNN in a 2D image and then utilizes a lightweight regression PointNet to estimate (x_c, y_c, z_c) and θ.

3.2.2. Local confidence generation

Due to the large monocular depth estimation error at the edges of the object vehicle, we assign a weight to each point within the object vehicle and call it the local confidence. Considering that the closer the point to the 3D center point, the smaller its depth estimation error, we utilize a 3D Gaussian function to model the local confidence of Pseudo-LiDAR. Therefore, the larger its confidence, the smaller its error, and vice versa. Since the sizes of the vehicles in the KITTI dataset are similar, we take the average length, width and height of each vehicle to determine whether a point is inside or outside the object vehicles.

The 3D coordinates of a point in a Pseudo-LiDAR point cloud are denoted by $p(x, y, z)$ in 3D space. To facilitate the calculation, we use the vehicle's head direction as the x-axis. Since the vehicle has a yaw angle of θ, we rotate the Pseudo-LiDAR coordinate system by θ degrees in a counterclockwise direction to obtain the new coordinate representation as follows:

$$\begin{cases} x^{'} = x\cos\theta + y\sin\theta \\ y^{'} = -x\sin\theta + y\cos\theta \\ z^{'} = z \end{cases} \tag{2}$$

where the new coordinates of $p(x, y, z)$ and 3D center point (x_c, y_c, z_c) are (x', y', z') and ($x_c^{'}$,$y_c^{'}$,$z_c^{'}$), respectively.

The 3D Gaussian weight of point p with respect to vehicle b is

$$\alpha_{(p,b)} = \begin{cases} s(x^{'}, y^{'}, z^{'}) & y(p,b) = 1, \\ 0 & y(p,b) = 0. \end{cases} \tag{3}$$

Here, $y(p, b) = 0$ is the point where p is not inside vehicle b. Then, the expression for the calculation of y is as follows:

$$y(p,b) = \begin{cases} 0 & if\left|x_c^{'} - x^{'}\right| \rangle l/2 \\ 0 & if\left|y_c^{'} - y^{'}\right| \rangle w/2 \\ 0 & if\left|z_c^{'} - z^{'}\right| \rangle h/2 \\ 1 & otherwise. \end{cases} \tag{4}$$

Here, ($x_c^{'}$,$y_c^{'}$,$z_c^{'}$) is the estimated 3D center point after the rotation transformation. l, w and h are the average length, width and height of b, respectively.

If point p is inside vehicle b, we calculate the confidence weights inside the vehicle by a 3D Gaussian function as follows:

$$s(x^{'}, y^{'}, z^{'}) = \frac{1}{(2\pi)^{\frac{3}{2}}\sigma^3} e^{-\frac{(x^{'}-x_c^{'})^2 + \left[(y^{'}-y_c^{'})\frac{l}{h}\right]^2 + \left[(z^{'}-z_c^{'})\frac{l}{w}\right]^2}{2\sigma^2}} \tag{5}$$

where σ is the decay rate parameter of the 3D Gaussian function.

Then, the expression of the local confidence $S_{Local}(p)$ of point p is

$$S_{Local}(p) = \max\left(\lambda_\alpha \cdot f_{norm}\left(\alpha_{(p,b)}\right), \xi_\alpha\right) \tag{6}$$

Here, λ_α is the weight balance parameter. f_{norm} is the weight normalized function, which ensures that the confidence of p is between 0 and 1. ξ_α is the background confidence threshold.

3.2.3. Global confidence generation

Lack of stereo visual information in monocular images leads to poor performance in monocular depth estimation of long-distance scenes, and the depth estimation error increases in the long-distance scenes. To solve this problem, we decrease the confidence of long-distance points with the confidence decay rate R_γ. Due to the diversity of autonomous driving scenes, the depth distribution varies greatly. Therefore, we calculate R_γ for different scenes as follows:

$$R_\gamma = \frac{1}{\lambda_\beta f_E(Q) + f_D(Q)} \tag{7}$$

Here, Q is a point cloud set of the current scene. λ_β is a global balance parameter to adjust the weight of the mean and variance of Q. $f_E(Q)$ and $f_D(Q)$ denote the mean and variance of Q, respectively.

$$f_E(Q) = \frac{\sum_{p\in Q} d_p}{|Q|}, f_D(Q) = \sqrt{\frac{\sum_{p\in Q}\left[d_p - f_E(Q)\right]}{|Q|}} \tag{8}$$

Here, d_p is the depth value of p. (7) shows that the mean and variance of Q are large, and a smaller decay rate is set to prevent the confidence from decaying too much. This setting avoids the object points in the scene being completely filtered out.

The confidence decay rate R_γ can make the confidence decrease as the scene distance increases, and the global confidence $S_{Global}(p)$ of point p is

$$S_{Global}(p) = \max\left(1 - R_\gamma d_p, \xi_\beta\right) \tag{9}$$

ξ_β is the global confidence background threshold to ensure that long-distance background points are not completely filtered out and long-distance object vehicle points are sampled.

The final confidence $S(p)$ of point p is generated by weighting the global confidence over the local confidence, i.e.,

$$S(p) = S_{Local}(p) \cdot S_{Global}(p) \tag{10}$$

3.2.4. Confidence sampling

Once the Pseudo-LiDAR confidence distribution is obtained, the sampled point cloud set Q_{sam} is obtained by sampling the raw Pseudo-LiDAR point cloud Q_{raw} as follows:

$$Q_{sam} = \{p | S(p) > rand(0,1), p \in Q_{raw}\} \tag{11}$$

rand(0, 1) is a function that can generate a random variable between 0 and 1. This equation shows that when the confidence of point p is greater than *rand*(0, 1), then point p is retained, and vice versa, p is filtered out. It can be seen that points with small confidence, such as background points, are filtered out.

3.3. Pseudo-LiDAR 3D object detection network with hierarchical geometric feature extraction

In this section, we take the PV-RCNN network as a 3D detector for our detection network. In PV-RCNN [18], its most critical point is to encode all voxel features of the entire scene into a small number of keypoints. However, this results in insensitivity to the geometric features of keypoints. To further enrich the feature information of keypoints in PV-RCNN, we designed a hierarchical geometric feature extraction

module. After the keypoints are extracted from the raw point cloud, we construct a local geometric neighborhood for each keypoint by using Local Attention Feature Encoding (LAFE) and then aggregate geometric neighborhood features to the keypoint. We also aggregate the corresponding RGB information to each keypoint to enrich the feature information of the keypoints. Finally, we design a dual transformer to capture the global geometric feature information between the keypoints.

3.3.1. Local attention feature encoding

The process of feature encoding is as follows. First, we assign the color information of each pixel to enrich the point cloud information. We combine the spatial information and color information. As shown in Fig. 3, we encode its spatial geometric information f_{xyz} (XYZ position of the point cloud) and color information f_{rgb} (RGB information of the point cloud) for each keypoint using the following equation:

$$f_e = MLP(f_{xyz}) \oplus MLP(f_{rgb}) \tag{12}$$

Here, f_e is the encoding feature, MLP is the multi-layer perceptron and $\oplus$ is the connection operation.

Then, we use the K-Nearest Neighbor (KNN) to find the neighbor points of the keypoints in the raw point cloud to construct a local neighborhood. Herein, KNN realizes clustering based on the Euclidean distance from the raw point to the keypoint. Inspired by Ref. [31], for keypoint i with K neighbor points, we encode its local neighborhood feature f_l^{ik} as

$$f_l^{ik} = MLP\left(f_{xyz}^{i} \oplus f_{xyz}^{i} - f_{xyz}^{ik} \oplus f_e^{i} \oplus f_e^{i} - f_e^{ik}\right) \tag{13}$$

where k denotes the k-th neighbor of the keypoint i. $f_{xyz}^{i} - f_{xyz}^{ik}$ and $f_e^{i} - f_e^{ik}$ denote the relative position (representing the geometric structural features between keypoints) and relative features of the k-th neighbor point, respectively. We enhance the geometric features of keypoints by combining the position information and color information from the raw point cloud.

Aggregating neighboring features by maximum or average pooling used in existing works results in irrelevant information around keypoints being retained and important information being lost. We introduced an attention mechanism to selectively aggregate beneficial features by learning weights. Inspired by Ref. [32], our attentional aggregation approach is as follows. The attention weight is calculated for each neighbor point of keypoint i as follows:

$$\alpha_i^k = softmax\left(MLP\left(f_{xyz}^{i} - f_{xyz}^{ik} \oplus f_e^{i} - f_e^{ik}\right)\right) \tag{14}$$

where *softmax* is a normalization function.

The final local neighborhood encoding feature F_{agg}^{i} of keypoint i can be calculated by the following equation:

$$F_{agg}^{i} = \sum_{k \in K} \alpha_i^k * f_l^{ik} \tag{15}$$

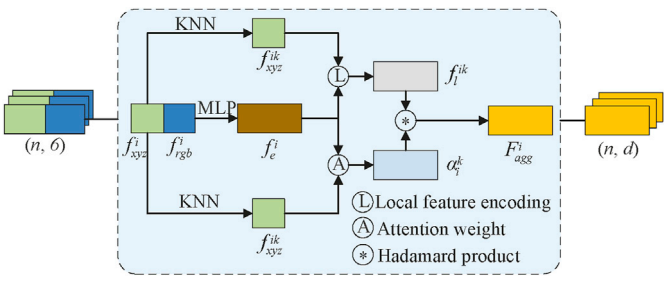


Fig. 3. Local attention feature encoding module. (MLP: Multi-Layer Perceptron, KNN: K-Nearest Neighbor.)

where K is the number of neighbors. * is the Hadamard product.

3.3.2. Global geometric feature extraction with dual transformer

Recently, transformers and self-attention have shown great potential in the field of NLP [33,34] and computer vision. Transformers have the advantage of capturing long-term correlations in sequences. Since point clouds are irregular data used to represent 3D spatial information, we consider them as a disordered sequence, and then apply transformers to capture the global correlation between points. There are two main types of self-attention: vector [35] and scalar [33]. Scalar attention focuses on the spatial correlation between points and obtains the long-range representation of contextual features. The attention weight of vector attention is a vector, so it can independently modulate each feature channel. Inspired by Ref. [36], we combine the advantages of these two types of attention and propose a dual transformer structure to enhance the global geometric feature representation of keypoints. The details of these two transformers are described as follows.

The computation of the transformer based on scalar attention [33] is finished by

$$\begin{cases} \mathbf{y}_{si} = \sum\limits_{\mathbf{x}_j \in \chi} \rho\left(\alpha(\mathbf{x}_i)^T \psi(\mathbf{x}_j) + \delta\right)\beta(\mathbf{x}_j) \\ \delta = \theta\left(\mathbf{p}_i^{\mathrm{T}} \mathbf{p}_j\right) \end{cases} \tag{16}$$

Here, χ is the set of features. $\mathbf{y}_{si}$ is the output feature of keypoint i. α, ψ and β is a feature transformation for points such as MLP. ρ is a normalization function such as *softmax*. δ is a position encoding function that is used to acquire the position relationship between point pairs to provide more contextual information. θ is an MLP, and $\mathbf{p}_i$ and $\mathbf{p}_j$ are the 3D coordinates of points i and j, respectively.

The differences between vector attention and scalar attention occur mainly in the calculation of attention weights and encoding functions. In vector attention, we adopt subtractive relations to calculate the attentional weights and positional encoding relations between pairs of points following [36]. The position encoding uses relative coordinates to better represent the geometric structure between the keypoints. The feature output $\mathbf{y}_{vi}$ of keypoint i is calculated as follows:

$$\begin{cases} \mathbf{y}_{vi} = \sum\limits_{\mathbf{x}_j \in \chi} \rho\left(\alpha(\mathbf{x}_j) - \psi(\mathbf{x}_j) + \delta\right)\beta(\mathbf{x}_j) \\ \delta = \theta\left(\mathbf{p}_i - \mathbf{p}_j\right) \end{cases} \tag{17}$$

Finally, the feature of keypoint i is $\mathbf{y}_f = \mathbf{y}_{si} \oplus \mathbf{y}_{vi}$, which fully exploits the advantages of the two attentions and improves the global correlation between the geometric structures of keypoints.

4. Experiment

4.1. Setup

4.1.1. Dataset

We verified our method on the popular KITTI [19]. The dataset provides images for training and testing. KITTI is currently one of the most important large-scale scenario datasets in the field of autonomous driving. However, the lack of diversity in time and weather is a limitation of KITTI. This dataset does not directly provide the ground truth of the test set, and specific requirements are needed for online testing. Therefore, we followed [12] to split the training set into two parts: 3712 samples (training set) and 3769 samples (validation set).

4.1.2. Evaluation metrics

We evaluated the results of both 3D and Bird's-Eye View (BEV) object detection on the KITTI validation set for cars to validate our algorithm. We calculate the corresponding Precision/Recall curves to obtain Average Precision (AP) with threshold values of 0.5 and 0.7. Although KTITI currently adopts AP_{40} instead of AP_{11}, many existing methods have

only the results of AP_{11}. Therefore, we use AP_{11} as our AP calculation criterion. AP_{BEV} is defined as the AP for BEV object detection, and AP_{3D} is the AP for 3D object detection. KITTI classifies the difficulty of the samples into three categories: easy, moderate, and hard.

4.1.3. Baseline

Our work is based on Pseudo-LiDAR from monocular images, so we compare it with the state-of-the-art 3D object detection methods that take monocular images as input. We obtained the AP_{11} of MonoDLE [38] according to the official code.[1]

4.2. Implementation

4.2.1. Pseudo-LiDAR generation

We perform depth estimation on 7481 images from KITTI by using the DORN to obtain the corresponding depth maps. The Pseudo-LiDAR obtained by depth map coordinate transformation and projection transformation is on the camera space. Therefore, we need to convert the visual point cloud to a Velodyne LiDAR point cloud by using the official camera matrix provided by KITTI.

4.2.2. Confidence sampling

When calculating the local confidence, λ_α (weight balance parameter) is 5 and ξ_α (background threshold) is 0.2 to ensure that the external 3D background points of the object vehicle are not completely filtered out. In calculating the global confidence, λ_β (global balance parameter) is 1.5, and ξ_β (background threshold) is 0.2 to ensure that the long-distance points are not completely filtered out. With the above parameter configuration, points with high confidence are retained by random sampling, and unimportant background points are filtered out.

4.2.3. Hierarchical geometric feature extraction

The confidence-sampled Pseudo-LiDAR is used as input for the PV-RCNN network. The number of samples n is 2048, so we optimized the features of these 2048 keypoints. K is set to 16. The dual transformer computation consumes GPU memory. Therefore, we sampled the keypoints to reduce memory usage. Finally, the sampled keypoints were restored to the raw number of keypoints by trilinear interpolation.

4.2.4. Training

We constructed our entire network framework based on the PV-RCNN network and trained it on an RTX2080Ti GPU. The Adam algorithm with a learning rate of 0.01 was used for training in 50 epochs.

4.3. Experiment results

4.3.1. Comparison with other methods

In Table 1, we summarize the detection results of our method and state-of-the-art methods on the KITTI validation set for cars. Our method outperforms the other methods on three different detection difficulty levels, i.e., easy, moderate and hard. For the BEV task, AP_{BEV} (in %) is improved by approximately 5.4 compared to AM3D at IoU = 0.7 (moderate). For the 3D detection task, AP_{3D} (in %) is improved by approximately 5.0 compared to MonoFLEX at IoU = 0.7 (moderate). This strongly demonstrates the importance of the quality of the generated Pseudo-LiDAR point clouds and the geometric structural features of the point cloud. Our method does not distinguish cars and pedestrians, and the performance of detection only depends on the estimation error of the object depth. Therefore, our method is also effective for pedestrians according to the detection performance for cars.

4.3.2. Comparing effect of regression parameters on detection performance

To investigate the effect of the regression parameters, i.e., center coordinates (x, y, z), length, width and height (l, w, h) of the 3D bounding box on detection performance, we counted the Mean Square Error (MSE) of the regression parameters and the number of 3D bounding boxes at different distances. The mean square error is obtained by the following formula:

$$\begin{cases} MSE_{xyz} = \frac{1}{3}\left((x_g - x_p)^2 + (y_g - y_p)^2 + (z_g - z_p)^2\right) \\ MSE_{lwh} = \frac{1}{3}\left((l_g - l_p)^2 + (w_g - w_p)^2 + (h_g - h_p)^2\right) \\ MSE = \alpha \mathrm{MSE}_{xyz} + \beta \mathrm{MSE}_{lwh} \end{cases} \tag{18}$$

Here, (x_g, y_g, z_g) are the 3D center coordinates, and (l_g, w_g, h_g) are the length, width and height of the real boxes, respectively. (x_p, y_p, z_p) are the 3D center coordinates, and (l_p, w_p, h_p) are the length, width and height of the predicted boxes, respectively. α and β are equilibrium coefficients of 20 and 10, respectively.

Figs. 4 (a) and (b) show that our method obtains almost the minimum (MSE) and the maximum number of bounding boxes at different distances. Although several methods have a small MSE of between 50 and 70 m, they can barely detect objects at long distances. Therefore, we know that a method with smaller (MSE) of the regression parameters can obtain better detection performance.

4.4. Ablation studies

4.4.1. Confidence sampling

To further demonstrate the reasonableness of the proposed point cloud confidence sampling method, we took an approach without confidence sampling or hierarchical feature extraction modules as the baseline. Table 2 lists the results of the ablation experiments. The model with a confidence sampling module improves by 2.0, 1.4 and 0.5 in the AP_{3D} (in %) values compared to the baseline on the three levels. Fig. 5 shows the detection results before and after confidence sampling, and there is not much difference between the detection results on the close scenes. However, the detection results improve when confidence sampling is executed on the long-distance scenes. From the visualization results and detection metrics, we can prove that our proposed method achieves a promising effect.

4.4.2. Hierarchical geometric feature extraction

In Table 3, we verify the detection performance of the Confidence Sampling Module (CSM) and the Hierarchical Geometric Feature Extraction Module (HGFEM) compared with the baseline. In both cases, the performance of 3D object detection is improved. Enhancing the geometric features of point clouds is a promising solution for monocular Pseudo-LiDAR 3D object detection.

4.4.3. Dual transformer structure

To further verify the effect of transformer structure on the performance of 3D object detection, we compared three different transformer structures, i.e., (1) dual transformer with a parallel structure (parallel TR), (2) dual transformer with a cascade structure (cascade TR) and (3) single transformer structure (single TR). For convenience, we removed the local attention feature encoding module in the HGFEM. In Table 4, we can see that the performance of the dual transformer is better than that of the single transformer, and the parallel TR performs best on moderate and hard samples but worse than the cascade TR on easy samples. Therefore, the parallel structure transformer is verified.

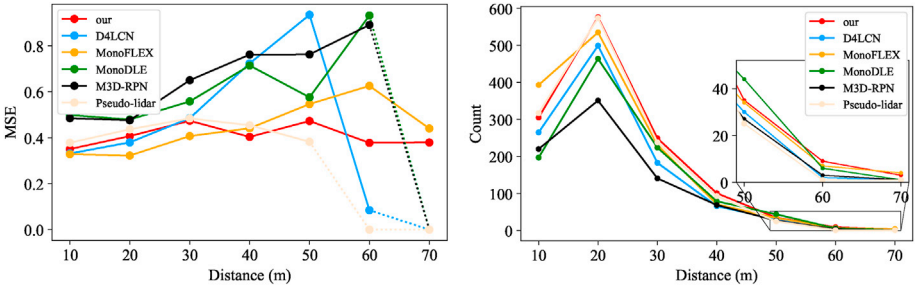
4.5. Visualization analysis

Fig. 6 shows the 3D visualization results of our methods. In most cases, our method can accurately predict the 3D bounding boxes in long-distance scenes and effectively handle the poor quality of the generated Pseudo-LiDAR point clouds. However, our approach has limitations since

[1] https://github.com/xinzhuma/monodle.

Table 1
Performance comparison on KITTI validation set.

Method	AP_{BEV}/AP_{3D} (in %), IoU = 0.5			AP_{BEV}/AP_{3D} (in %), IoU = 0.7		
	Easy	Moderate	Hard	Easy	Moderate	Hard
MonoGRNet [13]	-/50.51	-/36.97	-/30.82	-/13.88	-/10.19	-/7.62
M3D-RPN [14]	55.37/48.96	42.49/39.57	35.29/33.01	25.94/20.27	21.18/17.06	17.09/15.21
Pseudo-LiDAR [15]	70.8/66.3	49.4/42.3	42.7/38.5	40.6/28.2	26.3/18.5	22.9/16.4
AM3D [28]	72.64/68.86	51.82/49.19	44.21/42.24	43.75/32.23	28.39/21.09	23.87/17.26
D4LCN [37]	-	-	-	-/26.97	-/21.71	-/18.22
MonoDLE [38]	-	-	-	-/23.75	-/20.71	-/18.00
MonoFLEX [39]	-	-	-	-/28.17	-/21.92	-/19.07
Ours	**75.41/71.46**	**56.59/53.5**	**50.96/46.2**	**47.93/39.21**	**33.83/26.87**	**28.40/22.60**



(a) Mean square error (b) Number of 3D bounding boxes

Fig. 4. Mean square error of regression parameters (left) and number of 3D bounding boxes (right). (Dotted line indicates that no object was detected within that distance.)

Table 2
Ablation study for confidence sampling. Effects of using Local Confidence Module (+LCM) and Global Confidence Module (+GCM) are shown.

Method	IoU = 0.7, AP_{3D} (in %)		
	Easy	Moderate	Hard
baseline	35.24	24.55	20.75
+LCM	36.10	25.22	20.88
+GCM	36.72	25.30	21.06
+LCM + GCM	**37.28**	**25.92**	**21.28**

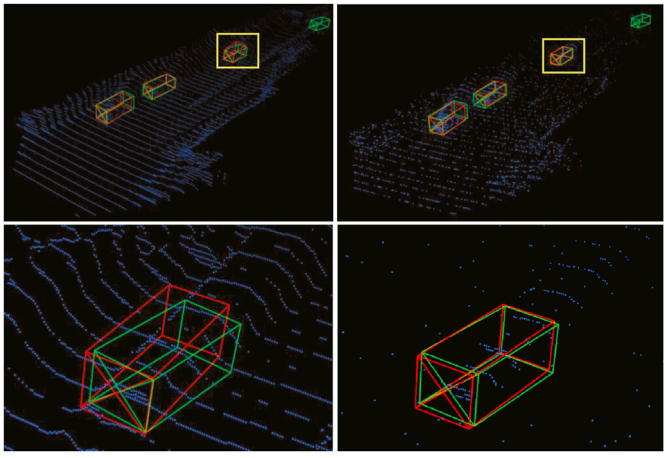


Fig. 5. 3D detection results before (left) and after (right) confidence sampling. 3D detection results for entire scene (top) and for local enlargement of 3D detection results in yellow box (bottom).

the generated Pseudo-LiDAR does not have the full geometry of the vehicle for the object being truncated in the image, e.g., the object in the yellow box. We will address this in the future.

Table 3
Ablation study with hierarchical geometric feature extraction. Effects of using Hierarchical Geometric Feature Extraction Module in baseline (+HGFEM) and in Confidence Sampling Module (+CSM + HGFEM) are shown.

Method	IoU = 0.7, AP_{3D} (in %)		
	Easy	Moderate	Hard
baseline	35.24	24.55	20.75
+HGFEM	37.23	25.45	21.16
+CSM + HGFEM	39.21	26.87	22.60

Table 4
Ablation study with dual transformer structure.

Method	IoU = 0.7, AP_{3D} (in %)		
	Easy	Moderate	Hard
single TR	34.26	25.58	20.87
cascade TR	**36.81**	25.93	20.86
parallel TR	36.61	**26.00**	**22.34**

5. Conclusion

In this paper, we presented a novel 3D object detection method by designing a confidence sampling strategy and a hierarchical geometric feature extraction module. We obtained a new benchmark for monocular 3D object detection by taking full advantage of the LiDAR-based 3D object detection method, i.e., PV-RCNN. The proposed Pseudo-LiDAR confidence sampling strategy was applied to process point clouds, addressing the problem of large depth estimation errors and improving the quality of Pseudo-LiDAR point clouds. The proposed hierarchical geometric feature extraction module uses attention and a transformer to extract the features of point clouds, and this can make the detection model more focused on the geometric features. Experiments verified that the detection performance of our method achieves significant advantages on KITTI (car) on easy, moderate and hard samples. However, there is

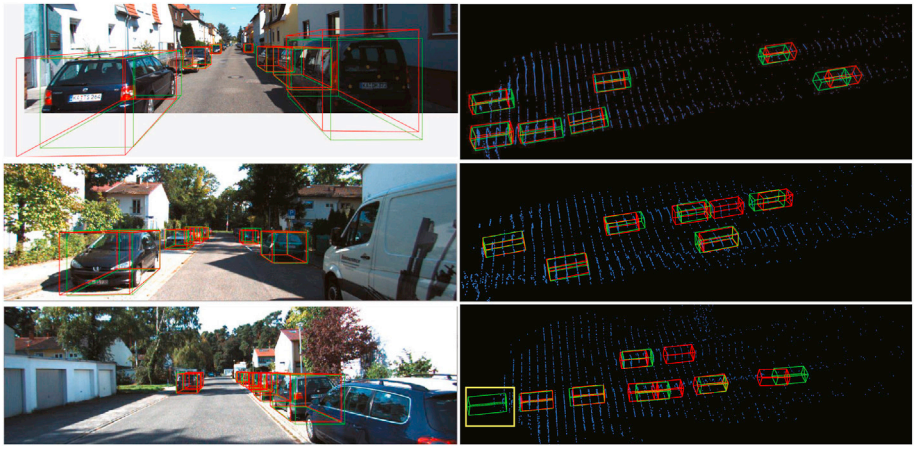


Fig. 6. 3D detection results for RGB image (left) and Pseudo-LiDAR (right). Green boxes: ground truth. Red boxes: predicted.

still room for improving the accuracy of monocular depth estimation information.

Declaration of competing interest

We declare that we have no conflict of interests in our manuscript.

Acknowledgements

This work was supported by the National Key Research and Development Program of China (2020YFB1807500), the National Natural Science Foundation of China (62072360, 62001357, 62172438, 61901367), the key research and development plan of Shaanxi province (2021ZDLGY02-09, 2023-GHZD-44, 2023-ZDLGY-54), the Natural Science Foundation of Guangdong Province of China (2022A1515010988), Key Project on Artificial Intelligence of Xi'an Science and Technology Plan (2022JH-RGZN-0003, 2022JH-RGZN-0103, 2022JH-CLCJ-0053), Xi'an Science and Technology Plan (20RGZN0005) and the Proof-of-concept fund from Hangzhou Research Institute of Xidian University (GNYZ2023QC0201).

References

[1] T. Qiu, Z. Zhao, T. Zhang, C. Chen, C.P. Chen, Underwater internet of things in smart ocean: system architecture and open issues, IEEE Trans. Ind. Inf. 16 (7) (2019) 4297–4307.
[2] W. Sun, S. Lei, L. Wang, Z. Liu, Y. Zhang, Adaptive federated learning and digital twin for industrial internet of things, IEEE Trans. Ind. Inf. 17 (8) (2020) 5605–5614.
[3] C. Wang, C. Chen, Q. Pei, Z. Jiang, S. Xu, An information centric in-network caching scheme for 5g-enabled internet of connected vehicles, IEEE Trans. Mobile Comput. 22 (6) (2023) 3137–3150.
[4] W. Sun, P. Wang, N. Xu, G. Wang, Y. Zhang, Dynamic digital twin and distributed incentives for resource allocation in aerial-assisted internet of vehicles, IEEE Internet Things J. 9 (8) (2022) 5839–5852.
[5] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.
[6] W. Sun, N. Xu, L. Wang, H. Zhang, Y. Zhang, Dynamic Digital Twin and Federated Learning with Incentives for Air-Ground Networks, IEEE Trans. Netw. Sci. Eng. 9 (1) (2022) 321–333.
[7] C. Chen, L. Liu, S. Wan, X. Hui, Q. Pei, Data dissemination for industry 4.0 applications in internet of vehicles based on short-term traffic prediction, ACM Trans. Internet Technol. 22 (1) (2021) 1–18.
[8] X. Chen, S. Leng, J. He, L. Zhou, Deep learning based intelligent inter-vehicle distance control for 6g-enabled cooperative autonomous driving, IEEE Internet Things J. 8 (20) (2021) 15180–15190.
[9] R. Girshick, Fast r-cnn, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 1440–1448.
[10] A. Simonelli, S.R. Bulo, L. Porzi, M. López-Antequera, P. Kontschieder, Disentangling monocular 3d object detection, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 1991–1999.
[11] B. Xu, Z. Chen, Multi-level fusion based 3d object detection from monocular images, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 2345–2353.
[12] X. Chen, K. Kundu, Z. Zhang, H. Ma, S. Fidler, R. Urtasun, Monocular 3d object detection for autonomous driving, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 2147–2156.
[13] Z. Qin, J. Wang, Y. Lu, Monogrnet: a geometric reasoning network for monocular 3d object localization, Proc. AAAI Conf. Artif. Intell. 33 (2019) 8851–8858.
[14] G. Brazil, X. Liu, M3d-rpn: monocular 3d region proposal network for object detection, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 9287–9296.
[15] Y. Wang, W.-L. Chao, D. Garg, B. Hariharan, M. Campbell, K.Q. Weinberger, Pseudo-lidar from visual depth estimation: bridging the gap in 3d object detection for autonomous driving, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 8445–8453.
[16] C.R. Qi, W. Liu, C. Wu, H. Su, L.J. Guibas, Frustum pointnets for 3d object detection from rgb-d data, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 918–927.
[17] C. Chen, Y. Zeng, H. Li, Y. Liu, S. Wan, A multi-hop task offloading decision model in mec-enabled internet of vehicles, IEEE Internet Things J. 10 (4) (2022) 3215–3230.
[18] S. Shi, C. Guo, L. Jiang, Z. Wang, J. Shi, X. Wang, H. Li, Pv-rcnn: point-voxel feature set abstraction for 3d object detection, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 10529–10538.
[19] A. Geiger, P. Lenz, R. Urtasun, Are we ready for autonomous driving? the kitti vision benchmark suite, in: 2012 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2012, pp. 3354–3361.
[20] C.R. Qi, H. Su, K. Mo, L.J. Guibas, Pointnet: deep learning on point sets for 3d classification and segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 652–660.
[21] S. Shi, X. Wang, H. Li, Pointrcnn: 3d object proposal generation and detection from point cloud, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 770–779.
[22] C. R. Qi, L. Yi, H. Su, L. J. Guibas, Pointnet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space, arXiv preprint arXiv:1706.02413.
[23] Y. Yan, Y. Mao, B. Li, Second: sparsely embedded convolutional detection, Sensors 18 (10) (2018) 3337.
[24] Y. Chen, S. Liu, X. Shen, J. Jia, Fast point r-cnn, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 9775–9784.
[25] C. Yan, E. Salman, Mono3d: open source cell library for monolithic 3-d integrated circuits, IEEE Trans Circ Syst I: Regul. Pap. 65 (3) (2017) 1075–1085.
[26] F. Chabot, M. Chaouch, J. Rabarisoa, C. Teuliere, T. Chateau, Deep manta: a coarse-to-fine many-task network for joint 2d and 3d vehicle analysis from monocular image, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 2040–2049.
[27] F. Manhardt, W. Kehl, A. Gaidon, Roi-10d: monocular lifting of 2d detection to 6d pose and metric shape, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 2069–2078.
[28] X. Ma, Z. Wang, H. Li, P. Zhang, W. Ouyang, X. Fan, Accurate monocular 3d object detection via color-embedded 3d reconstruction for autonomous driving, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 6851–6860.
[29] J. M. U. Vianney, S. Aich, B. Liu, Refinedmpl: Refined Monocular Pseudolidar for 3d Object Detection in Autonomous Driving, arXiv preprint arXiv:1911.09712.
[30] H. Fu, M. Gong, C. Wang, K. Batmanghelich, D. Tao, Deep ordinal regression network for monocular depth estimation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 2002–2011.
[31] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, A. Markham, Randla-net, Efficient semantic segmentation of large-scale point clouds, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 11108–11117.
[32] L. Wang, Y. Huang, Y. Hou, S. Zhang, J. Shan, Graph attention convolution for point cloud semantic segmentation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 10296–10305.

[33] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, in: Advances in Neural Information Processing Systems, 2017, pp. 5998–6008.
[34] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of Deep Bidirectional Transformers for Language Understanding, arXiv preprint arXiv: 1810.04805.
[35] H. Zhao, J. Jia, V. Koltun, Exploring self-attention for image recognition, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 10076–10085.
[36] H. Zhao, L. Jiang, J. Jia, P. Torr, V. Koltun, Point Transformer, arXiv preprint arXiv: 2012.09164.
[37] M. Ding, Y. Huo, H. Yi, Z. Wang, J. Shi, Z. Lu, P. Luo, Learning depth-guided convolutions for monocular 3d object detection, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, 2020, pp. 1000–1001.
[38] X. Ma, Y. Zhang, D. Xu, D. Zhou, S. Yi, H. Li, W. Ouyang, Delving into localization errors for monocular 3d object detection, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 4721–4730.
[39] Y. Zhang, J. Lu, J. Zhou, Objects are different: flexible monocular 3ds object detection, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 3289–3298.